⑲日本国特許庁（JP）　⑪特許出願公開

# ⑫公開特許公報（A）　平2−48732

| ⑤1Int. Cl.⁵ | 識別記号 | 庁内整理番号 | ㊸公開　平成2年(1990)2月19日 |
|---|---|---|---|
| G 06 F 9/38 | 310 F | 7361−5B | |

審査請求　有　請求項の数 3　（全7頁）

⑤4発明の名称　命令パイプライン方式のマイクロプロセッサ

㉑特　願　昭63−198789
㉒出　願　昭63(1988)8月11日

㉒発明者　岡本　光正　神奈川県川崎市幸区堀川町580番1号　株式会社東芝半導体システム技術センター内
㉑出願人　株式会社東芝　神奈川県川崎市幸区堀川町72番地
㉔代理人　弁理士　三好　保男　外1名

明　細　書

1．発明の名称

命令パイプライン方式のマイクロプロセッサ

2．特許請求の範囲

(1)　デコードされた命令を実行するため、オペランドアドレス計算部、アドレス変換部、オペランドフェッチ部、演算実行部および汎用レジスタ群を有する命令パイプライン方式のマイクロプロセッサにして、前記演算実行部を重複化して複数の命令を同時に実行できるように構成すると共に、前記汎用レジスタ群をプログラムの流れに従う処理データを格納するレジスタ群と、後続の命令を先回りして実行した結果のデータを格納するレジスタ群とで構成し、逐次的なプログラム処理に従う必要なしに処理可能な後続命令を先回りして実行可能にしたことを特徴とする命令パイプライン方式のマイクロプロセッサ。

(2)　前記汎用レジスタ群は、さらに後続の命令が先行する命令を飛びこして実行されているか否かを判定するための比較手段を有していることを特徴とする請求項1に記載の命令パイプライン方式のマイクロプロセッサ。

(3)　前記複数の各演算実行部は命令レジスタおよび演算手段とを備え、各命令レジスタ内のデータ先行を指定するデスティネーション・フィールドで指定された汎用レジスタ群の特定レジスタに格納されるようにしたことを特徴とする請求項1に記載の命令パイプライン方式のマイクロプロセッサ。

3．発明の詳細な説明

［発明の目的］

（産業上の利用分野）

本発明は命令並列実行方式のマイクロプロセッサ、特に命令を並列かつ迅速に実行しうる命令パイプライン方式のマイクロプロセッサに関するものである。

（従来の技術）

従来の命令パイプライン方式のマイクロプロセッサにおいては、加算命令などでオペランドのアドレス計算のために特定のレジスタが用いられる

が、その直前の転送命令などにより前記レジスタの内容が変更されるため、汎用レジスタへの書込ステージが終了するまで前記加算命令はオペランドアドレス計算ステージに移れないで命令の処理が遅れる問題があった。

すなわち、第７図は従来の命令パイプライン方式のマイクロプロセッサの概略構成を示す。

同図において、１はマイクロプロセッサＰと外部回路を接続するためのバス制御部（ＢＣＵ）、２は命令フェッチ部（ＩＦＵ）、３は命令を解読するためのデコーダ（ＤＥＣ）、４はオペランドアドレスを計算するためのオペランドアドレス計算部（ＯＡＧ）、５は論理アドレスを物理アドレスに変換するためのアドレス変換部（ＡＴ）、６はオペランドをフェッチするためのオペランドフェッチ部（ＯＰＦ）、７は命令を実行する演算実行部（ＥＸＵ）、８は複数のレジスタ$R_1$，$R_2$，$R_3$，$R_4$，…（図示せず）からなる汎用レジスタ群（ＧＲ）を示す。

第７図に示すマイクロプロセッサＰにより、例えば第６図に示すプログラム命令を実行した場合の命令パイプライン処理のタイミングを第８図に示す。第６図および第８図に示す如く、アドレスＡの内容を汎用レジスタ群８内の図示しないレジスタ$R_1$へ転送する命令$a_1$とし、汎用レジスタ群８内の$R_3$レジスタのデータを$R_2$へ転送する命令を$a_2$、上記レジスタ$R_2$で修飾される処のアドレスＢの内容をレジスタ$R_4$へ転送する加算命令を$a_3$、レジスタ$R_2$のデータをアドレスＣで示されるメモリへ転送する命令を$a_4$とすると、命令$a_3$でオペランドのアドレス計算を行なうためにレジスタ$R_2$の内容が用いられるが、命令$a_2$でレジスタ$R_2$が変更されてしまう。したがって命令$a_2$の汎用レジスタ群８への転送が（書込み）が終了するまで命令$a_3$はオペランドアドレス計算部へ入れず該命令の処理が遅れてしまう。

（発明が解決しようとする課題）

すなわち、従来のマイクロプロセッサにおいては演算実行部および汎用レジスタが重複化されていないために汎用レジスタ群８を更新する命令が完全に実行されてからでないと後続の命令は実行されえなかった。

したがって、従来の命令パイプライン方式のマイクロプロセッサにおいては処理の流れが停滞しパイプライン方式の利点を生かしきれていなかった。

そこで、本発明は、上記に鑑みてなされたものであり、その目的とするところは、先行する命令の実行結果を待たずに後続の命令を先回りして実行することができる命令パイプライン方式のマイクロプロセッサを提供することである。

〔発明の構成〕

（課題を解決するための手段）

上記目的を達成するために、本発明に従う命令パイプライン方式マイクロプロセッサにおいては、演算実行部をメモリオペランドをもたない命令を実行する第１の演算実行部と、メモリオペランドをもつ命令を実行する第２の演算実行部と、浮動小数点命令を実行する第３の演算実行部とで構成すると共に、汎用レジスタを、プログラムの流れに沿った処理データを記憶するレジスタ群と、処理可能な命令を先取り実行した結果を記憶するレジスタ群とで構成している。

（作用）

演算実行部を重複化すると共に汎用レジスタ群の構成を、プログラムの流れに沿った処理データを記憶するレジスタ群と、処理可能な命令を先取り実行した結果を記憶するレジスタ群とに重複化することにより、汎用レジスタ群のデータ更新を行ない、先行する命令の実行結果を待たずに後続の命令を先回りして実行し、マイクロプロセッサの性能を上げるようにしている。

（実施例）

第１図は本発明による命令パイプライン方式のマイクロプロセッサの原理構成図を示す。同図において、第７図のものと同一の構成要素は同じ参照番号で示してある。第１図の命令パイプライン方式マイクロプロセッサにおいては、演算実行部をメモリオペランドをもたない命令を実行する第１の演算実行部１０（ＳＥＰ）、メモリオペラ

ンドをもつ命令を実行する第２の演算実行部１１（ＩＥＰ）、浮動小数点命令を実行する第３の演算実行部１２（ＦＥＰ）で構成している。なお、１３は解読（デコード）された命令を前記各演算実行部へ送出する命令送出部、１４は浮動小数点レジスタを示す。

第１図に示す本発明によるマイクロプロセッサにおいては第６図に示す同じパイプライン命令に対して第２図に示すように実行する。

すなわち、毎クロックごとに命令がフェッチされ処理されてゆくが、命令$a_1$はメモリオペランドをもつので命令フェッチ部２→デコーダ３→命令送出部１３→オペランドアドレス部４→アドレス変換部５→オペランドフェッチ部６→第２の演算実行部１１と処理される。命令$a_2$はメモリオペランドをもたない（レジスタオペランドのみ）命令であるので、命令フェッチ部２→デコーダ３→命令送出部１３→第１の演算実行部１０と処理される。第２図に示すように命令$a_2$と$a_1$の実行が第５クロックおよび第６クロックで終了してしまうので汎用レジスタ群８´内のデータとプログラムの流れに不一致が生じるのを避けるために汎用レジスタ群８´は後述するようにプログラムの処理結果データを保持するレジスタ群と、命令の実行結果を一時的に先回りして保持するレジスタ群とに重複化して解決している。

次に、第３図を参照して、第１図の汎用レジスタ８´および第１，第２，第３の演算実行部１０，１１，１２の詳細な構成を示す。

第３図において、第１の演算実行部１０は第１の演算器２０と第１の命令レジスタ２１を、第２の演算実行部１１は第２の演算器２２と第２の命令レジスタ２３、および第３の演算実行部１２は第３の演算器２４と第３の命令レジスタ２５を有している。そして前記各命令レジスタ２１，２３，２５のフォーマットは第４図のようになっている。

すなわち、第４図においてＯＰは関連する演算器の演算動作指定フィールド、ＳＲはソースレジスタ指定フィールド、ＤＲはデスティネーションレジスタ指定フィールド、ＳＡ／ＩＤはソースオペランドアドレスまたはイミーディエトデータ保持フィールド、ＤＡはデスティネーションオペランドアドレス保持フィールド、ＰＣは各演算器で実行中の命令アドレス保持フィールドを示す。

第３図に戻って、本発明による汎用レジスタ群８´はプログラムに従って処理された結果のデータを保持するレジスタ群ＣＧＲｉ（ｉ＝１，２，３…ｎ）と命令が実行された演算器からの演算結果を格納するレジスタ群ＦＧＲｉ（ｉ＝１，２，３…ｎ）に分けられている。なお、第３図で３０は命令アドレス比較部を示し、これは汎用レジスタ群８´に含まれていると見なしてもよい。

第５図はレジスタ群ＦＧＲのフォーマットの詳細を示す、各３ビットからなるタグ部Ｆ，Ｉ，Ｓと各データを保持するデータ保持部ＦＧＲｉからなっている。

第３図に戻って、各命令レジスタ２１，２３，２５のＰＣフィールドは、後続の命令が先行する命令を飛越して実行されているかどうかを判定するために前記命令アドレス比較部３０へ入力され、前記各ＰＣフィールドを比較している。比較結果から第１の命令レジスタ２１のＰＣフィールドが最小でないと出力$C_1$が“１”に、第２の命令レジスタ２３のＰＣが最小でないと出力$C_2$が“１”、そして第３の命令レジスタ２５のＰＣが最小でないと出力$C_3$が“１”となるような出力が前記比較部３０から出される。

更に前記アドレス比較部３０の出力$C_4$（ＦＧＲ書込み信号）は命令を実行した演算部の命令レジスタのアドレスＰＣが最小でない場合に“１”となり、最小の場合に前記比較部３０の出力$C_5$（ＣＧＲ書込み信号）は“１”となるように構成されている。

また各演算部の演算終了時に、レジスタＦＧＲ中のタグＳは$C_4=1$，$C_1=1$のとき“１”にセットされ、Ｉタグは$C_2=1$のとき“１”に、そしてＦタグは$C_3=1$のときに“１”にセットされるようになっている。

各演算器２０，２２，２４の演算結果は、各命令レジスタ２１，２３，２５のＤＲフィールドで

指定されたデータ保持部ＦＧＲｉに格納されるようになっている。

各演算器２０，２２，２４の演算終了時に、$C_4-0$，$C_5-1$の場合、各演算結果は各命令レジスタのＤＲフィールドで指定されるデータ保持部ＦＧＲｉおよびＣＧＲｉに格納されると共に、Ｆ，Ｉ，Ｓタグのいずれかセットされている別のＦＧＲｉのデータは対応するＣＧＲｉに指定されるようになっている。

したがって第２図に示すように、動作時において、命令$a_2$が実行される第５クロックで、第３図に示す第１の命令レジスタ２１のＰＣフィールドが命令$a_2$のアドレスを有し、第２の命令レジスタ２３のＰＣフィールドは命令$a_1$のアドレスを有しているが、第３の命令レジスタ２５のＰＣフィールドには命令送出部１３から命令が送出されてないのでそのＰＣフィールドにはアドレスがない。したがって第３の演算実行部１２では何らの命令の実行も行なわれない。

一方、前記のように第１の命令レジスタ２１には$a_2$が入っているのでそのアドレスＰＣフィールドは、第２の命令レジスタ２３に入っている命令$a_1$のアドレスＰＣフィールドよりも大であるから$C_1-1$となり演算結果がＦＧＲ$_2$へ書込まれると共にタグＳが１にセットされる。

第６クロックにおいては、第２の演算部１１で命令$a_1$が実行されるが、このとき、第１の演算部１０内の命令レジスタ２１では上記のように命令$a_2$がすでに実行され終っていて命令がないので何らの書きもしない。したがって$C_4-0$、$C_5-1$となるのでＣＧＲ$_1$およびＦＧＲ$_1$に命令$a_1$の演算結果が格納されると共にタグＳ$_1$が１となっているＦＧＲ$_2$のデータがＣＧＲ$_2$へ転送され、プログラムの流れに沿った処理データとして用いられるように記憶される。

上記の動作から判るように、次の命令$a_3$の実効アドレスの算出に必要なデータは、第５クロックの終了時にＦＧＲ$_2$（Ｒ$_2$レジスタに対応する）から取出すことができるので命令$a_3$は、直ちに実効アドレス計算部４へ進むことができる。したがって、従来のこの種のマイクロプロセッサのように$a_1$，$a_2$の実行終了を待ってから命令$a_3$の処理に入るというような逐次的なプログラム処理に従う必要がない。

［発明の効果］

以上述べたように本発明においては、演算実行部を重複化して複数の命令を同時に実行できるように構成すると共に、汎用レジスタ群もプログラムの流れに従う処理データを格納するレジスタ群と、後続の命令を先回りして実行した結果のデータを格納するレジスタ群とを有するように重複化構成としている。

したがって本発明における命令パイプライン方式のマイクロプロセッサにおいては、先行する命令の実行結果を待たずに後続の命令を先回りして実行できるのでマイクロプロセッサの性能を著しく向上することができる。

４．図面の簡単な説明

第１図は、本発明による命令パイプライン方式のマイクロプロセッサの基本構成図、

第２図は、第１図のマイクロプロセッサを構成する各部の命令処理流れ図、

第３図は、第１図の演算実行部および汎用レジスタ群の詳細な構成図、

第４図は、第３図の各命令レジスタのフォーマット図、

第５図は、汎用レジスタ群内の命令を先回り処理した結果を格納するレジスタ群ＦＧＲの入力フォーマット、

第６図は、従来のマイクロプロセッサの有する問題点を明らかにするプログラム例、

第７図は、従来技術による命令パイプライン方式のマイクロプロセッサの構成図、

第８図は、第７図のプロセッサ内の命令処理の流れ図、をそれぞれ示す。

１…バス制御部、

２…命令フェッチ部、

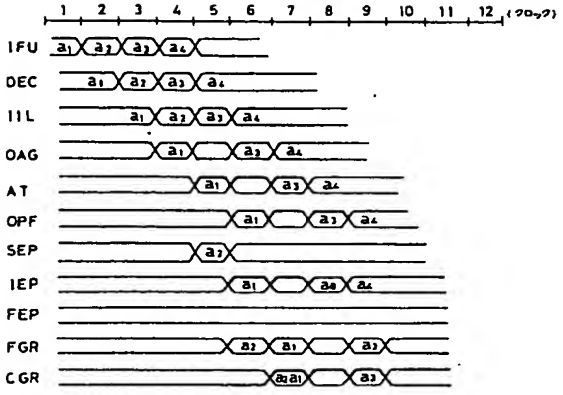
３…デコーダ、

４…オペランドアドレス計算部、

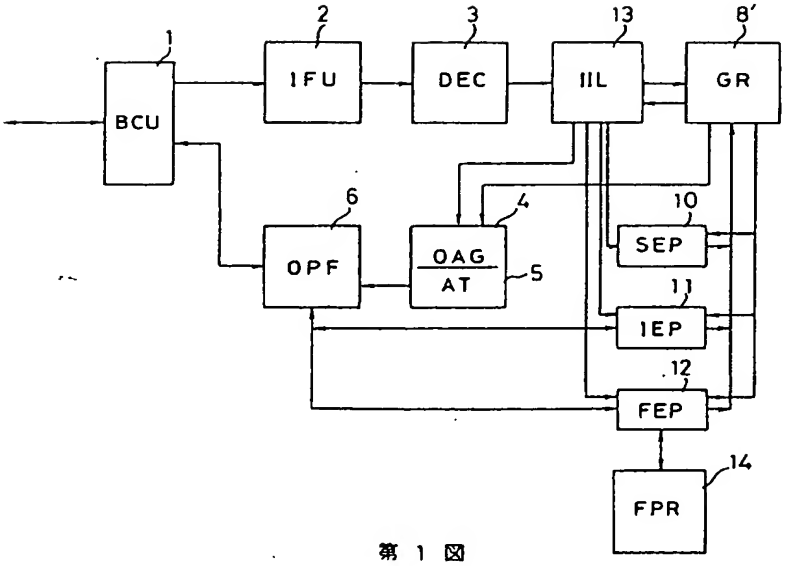
５…論理アドレス変換部、

6…オペランドフェッチ部、
8′…汎用レジスタ群、
10…第1の演算実行部、
11…第2の演算実行部、
12…第3の演算実行部、
13…命令送出部、
14…浮動小数点レジスタ。

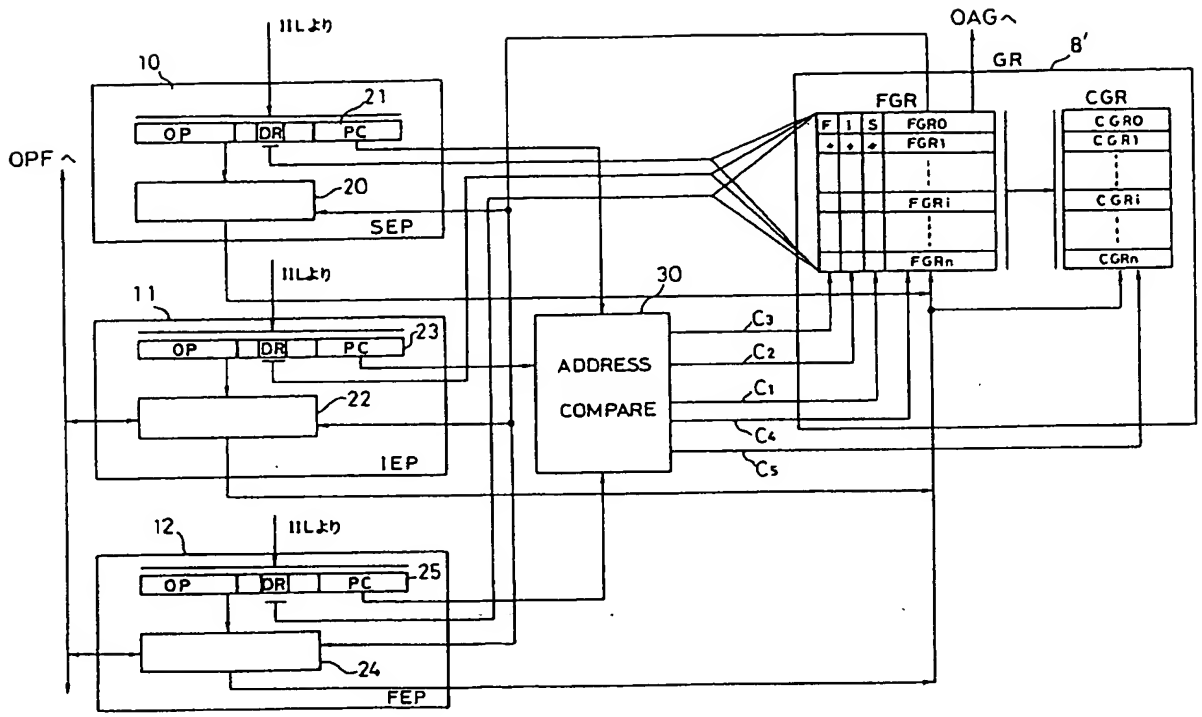
代理人弁理士 三好保男



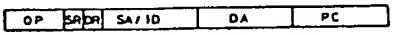

第2図




第1図




第 3 図

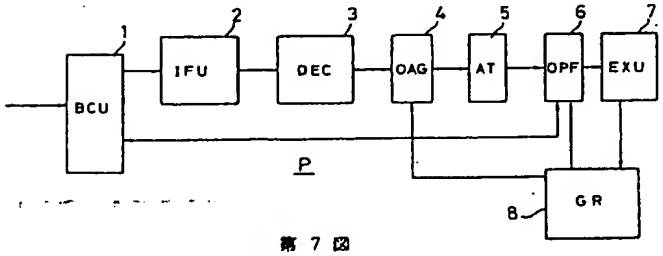



第 4 図

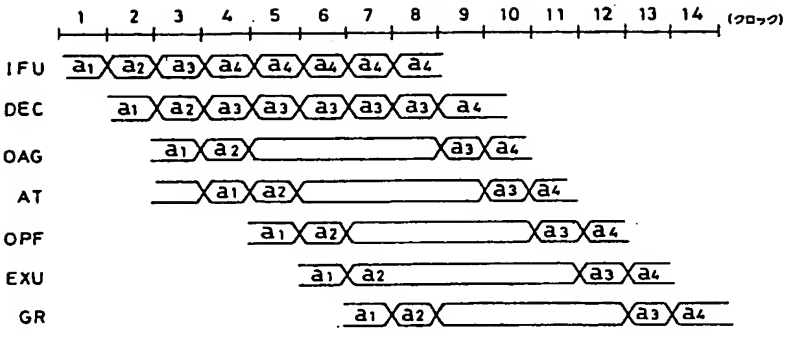

第 5 図

```
a1------MOV    adrs A , R1      ; adrs A ―R1
a2------MOV    R3 , R2          ;   R3―― R2
a3------ADD    adrs B (R2) , R4  ; adrs B(R2) + R4 ―R4
a4------MOV    R2 , adrs C      ;   R2 ―― adrs C
```

第 6 図




第 7 図




第 8 図

(19) Japan Patent Office (JPO)
(11) Publication Number
(12) Publication of Unexamined Patent Application (A) H2-48732
(51) International Classification[5] G 06 F 9/38
Identification Number: 310 F
JPO file number: 7361-5B
(43) Date of Publication of Application: 19 February 1990
Request for examination Not requested Number of Claims: 3 (total 7 pages)
(54) Title of the invention
Micro Processor of Instruction Pipeline System
(21) Application filing No. 63-198789
(22) Application filing date August 11, 1988
(72) Inventor(s) OKAMOTO MITSUMASA; TOSHIBA CORPORATION, SEMICONDUCTOR SYSTEM ENGINEERING DIVISION; 580-1, Horikawa-cho, Saiwai-ku, Kawasaki, Kanagawa, Japan
(71) Applicant(s) TOSHIBA CORPORATION; 72, Horikawa-cho, Saiwai-ku, Kawasaki, Kanagawa, Japan
(74) Agency MIYOSHI YASUO, Patent Attorney and & 1 other

Specification

1. Title of the Invention

MICRO PROCESSOR OF INSTRUCTION PIPELINE SYSTEM

2. What is Claimed is:

[Claim 1] A micro processor of instruction pipeline system, comprising: an operand address generator; an address transducer; an operand fetch unit; and an operation execution unit and a general-purpose register group in order to execute decoded instructions, wherein said operation execution unit is configured to be duplicated and simultaneously execute a plurality of instructions; and said general-purpose group is composed of a register group for storing processed data in accordance with the flow of a program and a register group for storing data of a result obtained by executing a subsequent instruction in advance, thereby enabling the processable subsequent instruction to be executed in advance without following sequential program processing.

[Claim 2] The micro processor of instruction pipeline system according to Claim

1, wherein said general-purpose register group further comprises a comparing means for judging whether or not the subsequent instruction is executed in advance of a preceding instruction.

[Claim 3] The micro processor of instruction pipeline system according to Claim 1, wherein each of a plurality of said operation execution unit comprises an instruction register and an operation means so as to perform storage in a specific register of the general-purpose register group specified in a destination field of the instruction register that specifies data precedence.

3. Detailed Description of the Invention

[Object of the Invention]

(Field of Industrial Application)

The present invention relates to a micro processor of instruction parallel execution system and more particularly to a micro processor of instruction pipeline system which can execute instructions in parallel and fast.

(Description of the Related Art)

In a conventional micro processor of instruction pipeline system, a specific register has been used for operand address calculation in an addition instruction or the like, but there has been a problem that since the contents of said register are modified by the last transfer instruction or the like, said addition instruction cannot transit to an operand address calculation stage until a write stage to a general-purpose register is finished, causing the processing of the instruction to delay.

Specifically, Fig. 7 shows a schematic configuration of a conventional micro processor of instruction pipeline system.

In this figure, a reference numeral 1 denotes a bus control unit (BCU) for connecting a micro processor P and an external circuit, a reference numeral 2 denotes an instruction fetch unit (IFU), a reference numeral 3 denotes a decoder (DEC) for decoding an instruction, a reference numeral 4 denotes an operand address generator (OAG) for calculating an operand address, a reference number 5 denotes an address transducer (AT) for transducing a logical address to a physical address, a reference numeral 6 denotes an operand fetch unit (OPF) for fetching an operand, a reference numeral 7 denotes an operation execution unit (EXU) for executing an instruction, and a reference numeral 8 denotes a general-purpose register group (GR) consisting of a plurality of registers $R_1$, $R_2$, $R_3$, $R_4$, ... (not shown).

Fig. 8 shows the timing of instruction pipeline processing in the case where, for example, program instructions shown in Fig. 6 are executed by the micro processor P shown in Fig. 7. As shown in Figs. 6 and 8, suppose an instruction to transfer contents of an address A to the register $R_1$ in the general-purpose register group 8 (not shown) is indicated by a reference sign $a_1$, an instruction to transfer data of the register $R_3$ to $R_2$ in the general-purpose register group 8 is indicated by a reference sign $a_2$, an addition instruction to transfer contents of an address B modified in the above-mentioned register $R_2$ to the register $R_4$ is indicated by a reference sign $a_3$, and an instruction to transfer data of the register $R_2$ to a memory indicated by an address C is indicated by a reference sign $a_4$. At this time, although the contents of the register $R_2$ are used to

calculate the operand address by the instruction $a_3$, the register $R_2$ is changed by the instruction $a_2$. Accordingly, the instruction $a_3$ cannot enter the operand address generator until the transfer of the instruction $a_2$ to the general-purpose register group 8 (write) is finished, which delays the processing of the instruction $a_3$.

(Problems to be Solved by the Invention)

In other words, in the conventional micro processor, since the operation execution unit and the general-purpose registers are not duplicated, a subsequent instruction cannot be executed before an instruction which updates the general-purpose register group 8 is completely executed.

Accordingly, in the conventional micro processor of instruction pipeline system, its processing flow stagnates and advantages of the pipeline system have not been exerted sufficiently.

The present invention seeks to solve the above-mentioned problem. The object is to provide a micro processor of instruction pipeline system capable of executing a subsequent instruction in advance without waiting for an execution result of a preceding instruction.

[Constitution of the Invention]

(Means for Solving the Problem)

In order to achieve the above-mentioned object, in a micro processor of instruction pipeline system according to the present invention, an operation execution unit is composed of a first operation execution unit for executing an instruction with no memory operand, a second operation execution unit for executing an instruction with a memory operand, and a third operation execution unit for executing a floating-point instruction, and general-purpose registers are composed of a register group for storing processed data in accordance with the flow of a program and a register group for storing a result obtained by executing a processable instruction in advance.

(Action)

The operation execution unit is duplicated and the configuration of the general-purpose register group is also duplicated by consisting of the register group for storing the processed data in accordance with the flow of the program and the register group for storing the result obtained by executing the processable instruction in advance to thereby update the data in the general-purpose register group and to executing the subsequent instruction in advance without waiting for the execution

result of the preceding instruction, which improves performance of the micro processor.

(Preferred Embodiment of the Invention)

Fig. 1 is a fundamental block diagram showing a micro processor of instruction pipeline system according to the present invention. In this figure, like components are indicated by the same reference numerals and signs as those in Fig. 7. In the micro processor of instruction pipeline system in Fig. 1, the operation execution unit is composed of a first operation execution unit 10 (SEP) for executing an instruction with no memory operand, a second operation execution unit 11 (IEP) for executing an instruction with a memory operand, and a third operation execution unit 12 (FEP) for executing a floating-point instruction. Incidentally, a reference numeral 13 denotes an instruction delivery unit for delivering a decoded instruction to each of said operation execution units, and a reference numeral 14 denotes a floating-point register.

In the micro processor according to the present invention as shown in Fig. 1, the same pipeline instructions as those of Fig. 6 are executed as shown in Fig. 2.

Specifically, an instruction is fetched at every clock to be processed, and the instruction $a_1$, having a memory operand, is processed from the instruction fetch unit 2, via the decoder 3, the instruction delivery unit 13, the operand address generator 4, the address transducer 5, and the operand fetch unit 6, to the second operation executing unit 11. The instruction $a_2$, having no memory operand (having only a register operand), is processed from the instruction fetch unit 2, via the decoder 3, and the instruction delivery unit 13, to the first operation execution unit 10. As shown in Fig. 2, the executions of the instructions $a_2$ and $a_1$ are finished in the fifth clock and the sixth clock and, therefore, in order to prevent discrepancy between data within a general-purpose register group 8' and the flow of a program, the general-purpose register group 8' is duplicated by consisting of a register group holding processing result data of the program and a register group temporarily holding an execution result by executing an instruction in advance as shown described later.

Next, referring to Fig. 3, the configurations of the general-purpose register 8' and the first, second, third operation execution units 10, 11 and 12 in Fig. 1 are described in detail.

In Fig. 3, the first operation execution unit 10 has a first operator 20 and a first instruction register 21, the second operation execution unit 11 has a second operator 22 and a second instruction register 23, and the third operation execution unit 12 has a third operator 24 and a third instruction register 25. The format of each of said instruction registers 21, 23 and 25 is as shown in Fig. 4.

Specifically, in Fig. 4, OP denotes an operation behavior specifying field of the relevant operator, SR denotes a source register specifying field, DR denotes a destination register specifying field, SA/ID donates a source operand address or immediate data holding field, DA denotes a destination operand address holding field, and PC denotes an instruction address holding field for an instruction being executed in each operator.

Back to Fig. 3, the general-purpose register group 8' according to the present invention is divided into a register group CGRi (i = 1, 2, 3...n) holding data of a result processed in accordance with a program and a register group FGRi (i = 1, 2, 3...n) storing an operation result from an operator executing an instruction. Incidentally, in Fig. 3, a reference numeral 30 denotes an instruction address comparing unit, which may be considered to be included the general-purpose register group 8'.

Fig. 5 shows the format of the register group FGR in detail. The register group FGR consists of a tag units F, I, and S each consisting of 3 bits, and a data holding unit FGRi for holding each data.

Back to Fig. 3, the PC fields of the respective instruction registers 21, 23 and 25 are inputted to said instruction address comparing unit 30 and said respective PC fields are compared to judge whether a subsequent instruction is executed in advance of a preceding instruction. Said comparing unit 30 outputs in such a manner based on the comparison result, that if the PC field of the first instruction register 21 is not the least, an output $C_1$ becomes "1", if the PC field of the second instruction register 23 is not the least, an output $C_2$ becomes "1", and if the PC field of the third instruction register 25 is not the least, an output $C_3$ becomes "1".

Furthermore, an output $C_4$ (an FGR write signal) of said address comparing unit 30 becomes "1" in the case where the address PC of the instruction register of an operator executing an instruction is not the least, while an output $C_5$ (a CGR write signal) of said comparing unit 30 becomes "1" in the case where said address PC is the least.

In addition, when the operation in each operator is finished, the tag S in the register FGR is designed to be set to "1" in the case of $C_4 = 1$ and $C_1 = 1$, the tag I to "1" in the case of $C_2 = 1$, and the tag F to "1" in the case of $C_3 = 1$.

The operation results of the respective operators 20, 22 and 24 are designed to be stored in the data holding units FGRi specified in the DR fields of the respective instruction registers 21, 23 and 25.

When the operations in the respective operators 20, 22 and 24 are finished, in the case of $C_4 = 0$ and $C_5 = 1$, operation results are designed to be stored in the data

holding units FGRi and CGRi specified in the DR fields of the instruction registers, respectively, and the other data of FGRi with any of the tags F, I and S set is designed to be specified to the corresponding CGRi.

Accordingly, as shown in Fig. 2, during operation, at the fifth clock when the instruction $a_2$ is executed, the PC field of the first instruction register 21 shown in Fig. 3 has the address of the instruction $a_2$, the PC field of the second instruction register 23 has the address of the instruction $a_1$, and the PC field of the third instruction register 25 has no address since the instruction delivery unit 13 has not delivered any instruction. Accordingly, the third operation execution unit 12 does not execute any instruction.

On the other hand, as mentioned above, since the first instruction register 21 contains $a_2$, its address PC field is more larger than the address PC field of the instruction $a_1$ contained in the second instruction register 23, $C_1$ becomes 1, so that the operation result is written to $FGR_2$ and the tag S is set to 1.

At the sixth clock, the instruction $a_1$ is executed in the second operation execution unit 11. At this time, the instruction register 21 in the first operation execution unit 10 has no instruction as the execution of the instruction $a_2$ has been finished and therefore, takes no action. Accordingly, $C_4$ is set to 0 and $C_5$ is set to 1, and the operation result of the instruction $a_1$ is stored in $CGR_1$ and $FGR_1$ and the data of $FGR_2$ with the tag $S_1$ set to 1 is transferred to and stored in $CGR_2$ so as to be used as processed data in accordance with the flow of the program.

As understood in the above-mentioned operations, since the data necessary for calculating the effective address of the next instruction $a_3$ can be derived from the $FGR_2$ (corresponding to the $R_2$ register) at the end of the fifth clock, the instruction $a_3$ can immediately proceed to the effective address generator 4. Accordingly, it is not necessary to follow the sequential program processing as in this type of conventional micro processor in which the instruction $a_3$ starts to be processed after the executions of $a_1$ and $a_2$ are finished.

[Effects of the Invention]

As described above, according to the present invention, the operation execution unit is configured to be duplicated and to simultaneously execute a plurality of instructions, and the general-purpose register group is also configured to be duplicated by consisting of the register group for storing processed data in accordance with the flow of a program and the register group for storing data of a result by executing a subsequent instruction in advance.

Accordingly, in the micro processor of instruction pipeline system according to

the present invention, since the subsequent instruction can be executed in advance without waiting for the execution result of the preceding instruction, the performance of the micro processor can be remarkably improved.

4. Brief Description of the Drawings

[Fig. 1]

Fig. 1 is a fundamental block diagram showing a micro processor of instruction pipeline system according to the present invention.

[Fig. 2]

Fig. 2 is a flow diagram showing instruction processing of respective units composing the micro processor of Fig. 1.

[Fig. 3]

Fig. 3 is a detailed block diagram showing an operation execution unit and a general-purpose register group of Fig. 1.

[Fig. 4]

Fig. 4 is a format diagram of each instruction register of Fig. 3.

[Fig. 5]

Fig. 5 shows an input format of a register group FGR for storing a result obtained by processing an instruction in advance in the general-purpose register group.

[Fig. 6]

Fig. 6 shows a program example clarifying a problem that a conventional micro processor has.

[Fig. 7]

Fig. 7 is a block diagram showing a conventional micro processor of instruction pipeline system according to the related art.

[Fig. 8]

Fig. 8 is a flow diagram showing instruction processing in the processor of Fig. 7.

1 bus control unit
2 instruction fetch unit
3 decoder
4 operand address generator
5 logical address transducer
6 operand fetch unit
8' general-purpose register group

10 first operation execution unit
11 second operation execution unit
12 third operation execution unit
13 instruction delivery unit
14 floating-point register

Patent agency, MIYOSHI YASUO